Manuel d'utilisation pour les laboratoires IEM

Auteur: G. Boutillier / C. Bardet

Objectif

Cette documentation donne de brèves explications sur l'utilisation des différents outils, environnements, protocoles utilisés dans le laboratoire de sytèmes embarqués. Ce document est un résumé, il ne remplace en aucun cas les différentes documentations fournies au début du laboratoire.

1. Carte de laboratoire EMB

La carte de laboratoire EMB est équipée d'une carte Freescale CSB535FS constituée d'un microcontrôleur i.MX21 (ARM926EJ-S). Elle possède 64Mo de mémoire SDRAM et 8Mo de mémoire FLASH. Une carte périphérique additionnelle est connectée à la carte CPU, elle possède un écran tactile, différents ports (UART, Ethernet, USB host/device), des connecteurs pour l'utilisation d'une carte Compact Flash et SD-Card, des entrées/sorties audio, des boutons et des LEDs disponibles à travers un périphériques I^2C, etc. Pour les laboratoires une carte interface a été rajoutée, elle possède 2 boutons et 2 LEDs directement connectés sur des GPIOs du i.MX21, ainsi que différents systèmes pour la génération d'horloges.

L'adresse IP de la carte EMB est **10.0.0.2**.

Connexion à la carte EMB depuis une machine Compaq:

- Alimenter la carte en 12V avec des fiches bananes depuis une alimentation du laboratoire
- Connexion du port série **B** de la machine hôte à la cible
- Connexion Ethernet de la machine hôte à la cible (carte réseau **2**, câble Ethernet croisée rouge)

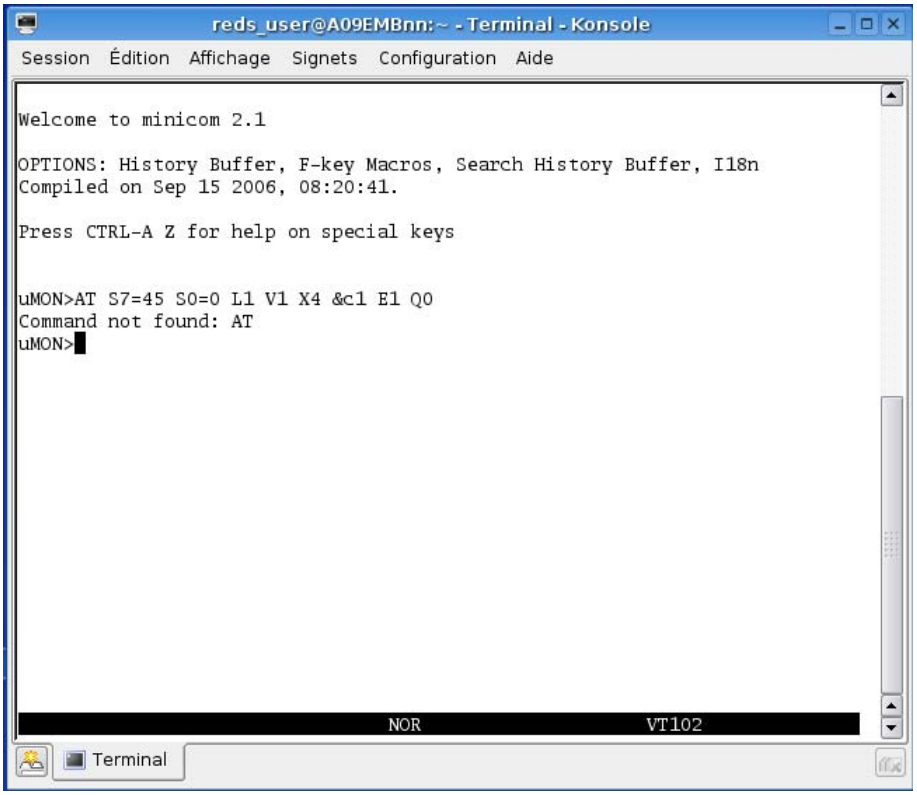
2. Minicom

Minicom est un programme utilisé dans ce laboratoire pour la connexion via le port série au moniteur ou au Linux embarqué sur la carte CSB535 depuis le Linux de la machine hôte.

Connexion:

Taper dans une console sur la machine hôte :

$ minicom


reds_user@A09EMBnn:~ - Terminal - Konsole
Session Édition Affichage Signets Configuration Aide
Welcome to minicom 2.1
OPTIONS: History Buffer, F-key Macros, Search History Buffer, I18n
Compiled on Sep 15 2006, 08:20:41.
Press CTRL-A Z for help on special keys
uMON>AT S7=45 S0=0 L1 V1 X4 &c1 E1 Q0
Command not found: AT
uMON>
NOR VT102
Terminal

Vous êtes maintenant connecté à votre cible!

3. Micro-monitor (uMON)

Micro-monitor (ou micro-moniteur) est le moniteur installé sur la carte CSB535. Il est stocké en flash mais exécuté en RAM. Il possède des commandes qui lui sont propres. Les commandes spécifiques à la gestion du système de fichiers TFS - *Tiny File System* - en *flash* sont similaires à celles de Linux. Pour ces dernières, il suffit de débuter votre commande par *tfs* suivi de la commande standard linux. Exemple: ***tfs ls***

4. Machine hôte Compaq

Les machines Compaq ont été configurées afin de disposer de tous les outils, environnements, services dont vous avez besoin pour le laboratoire de systèmes embarqués. Elles possèdent deux cartes réseau Ethernet : une carte pour la connexion au réseau local EMB, la deuxième pour la connexion à votre cible. L'adresse IP de cette deuxième carte réseau est **10.0.0.1**.

Pour transférer un fichier en flash sur la cible (scripts, exécutable, etc), vous devez taper la commande suivante sur une console de la machine hôte :

$ ttftp 10.0.0.2 **put** *nom_fichier_hôte nom_fichier_cible*

Pour transférer un fichier de la cible à la machine hôte, taper dans une console hôte :

$ ttftp 10.0.0.2 **get** *nom_fichier_cible nom_fichier_hôte*

Note importante : Si vous transférer un fichier exécutable, vous devez l'indiquer en ajoutant *,e* après le *nom_fichier_cible.*

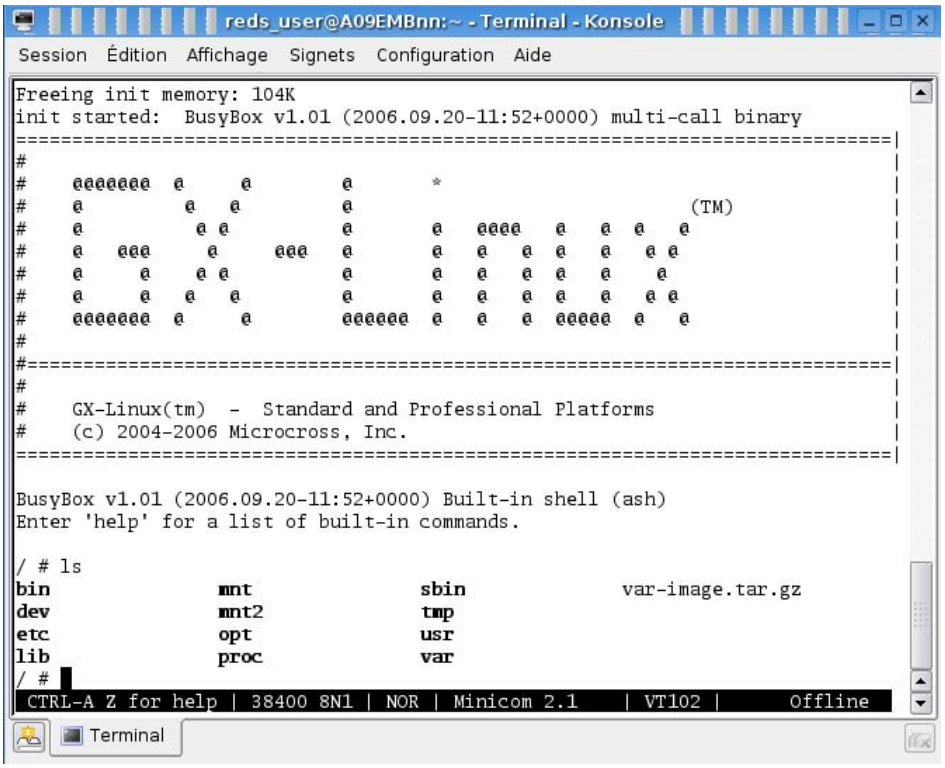
Exemple: $ ttftp 10.0.0.2 put helloworld helloworld,e

5. Linux embarqué

Pour démarrer Linux embarqué, il faut avoir transférer en flash le noyau Linux embarqué (zImage), l'arborescence racine compressé (romfs.img) et le script de démarrage *startlinux*. Utiliser la commande décrite ci-dessus pour réaliser ces différents transferts sur la cible.

1. Pour démarrer Linux, lancer depuis le micro-monitor le script *startlinux*.

uMON : startlinux


reds_user@A09EMBnn:~ - Terminal - Konsole
Session Édition Affichage Signets Configuration Aide
Freeing init memory: 104K
init started: BusyBox v1.01 (2006.09.20-11:52+0000) multi-call binary
GX-Linux(tm) - Standard and Professional Platforms
(c) 2004-2006 Microcross, Inc.
BusyBox v1.01 (2006.09.20-11:52+0000) Built-in shell (ash)
Enter 'help' for a list of built-in commands.
/ # ls
bin mnt sbin var-image.tar.gz
dev mnt2 tmp
etc opt usr
lib proc var
/ #
CTRL-A Z for help | 38400 8N1 | NOR | Minicom 2.1 | VT102 | Offline
Terminal

Vous avez maintenant démarré linux sous la cible, tester en essayant la commande ***ls***.

2. Transférer une application s'exécutant sous Linux embarqué sur la cible :

Machine hôte : Copier l'exécutable que vous désirez transférer sur la cible dans le répertoire /share ou dans le répertoire *share_freescale* placé sur le bureau.

Cible : Aller dans le répertoire /var, créer un répertoire portant votre nom. Depuis ce même répertoire taper dans la console :

$ tftp –g –r *nom_fichier* 10.0.0.1

Puis changer les droits de votre fichier pour pouvoir l'exécuter :

$ chmod 777 *nom_fichier*

Pour récupérer un fichier de la cible depuis la machine hôte, taper dans une console :

$ tftp –p –l *nom_fichier* 10.0.0.1

6. Qemu

Qemu est un programme permettant d'émuler un autre environnement. Ce programme est très intéressant pour pouvoir tester des aplications embarquées sans pour autant avoir besoin de charger les exécutables sur une carte. Il est également possible d'émuller un autre environnemnt depuis Linux (par ex Windows) ce qui peut être pratique pour utiliser certain programme.

Sur les machines compaq Qemu a été compilé pour les systems ARM.

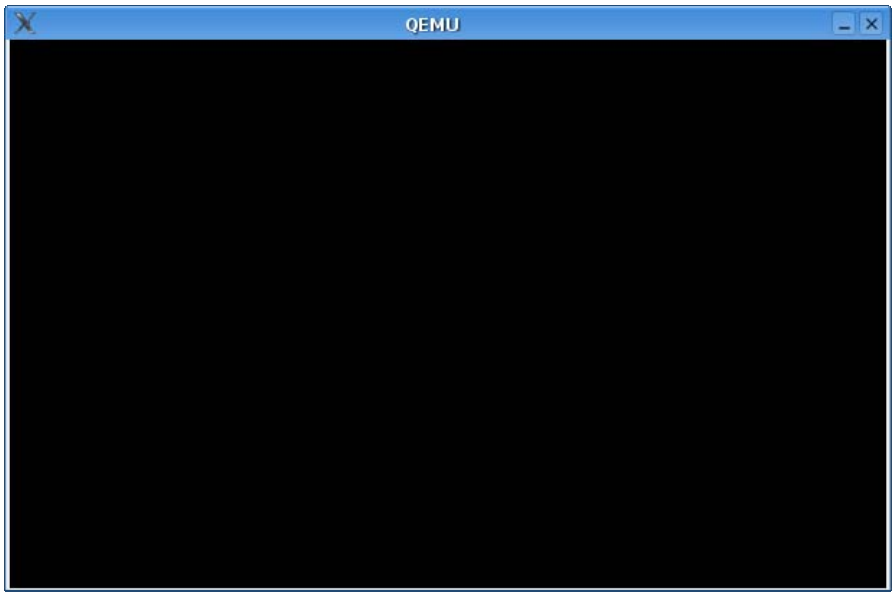
Qemu s'exécute avec la commade suivante:

$ qemu-system-arm *[suivit de divers paramètres]*

Pour lister les paramètres:

$ qemu-system-arm –help ou simplement **$ qemu-sytem-arm**

Qemu s'ouvre dans une nouvelle fenêtre:

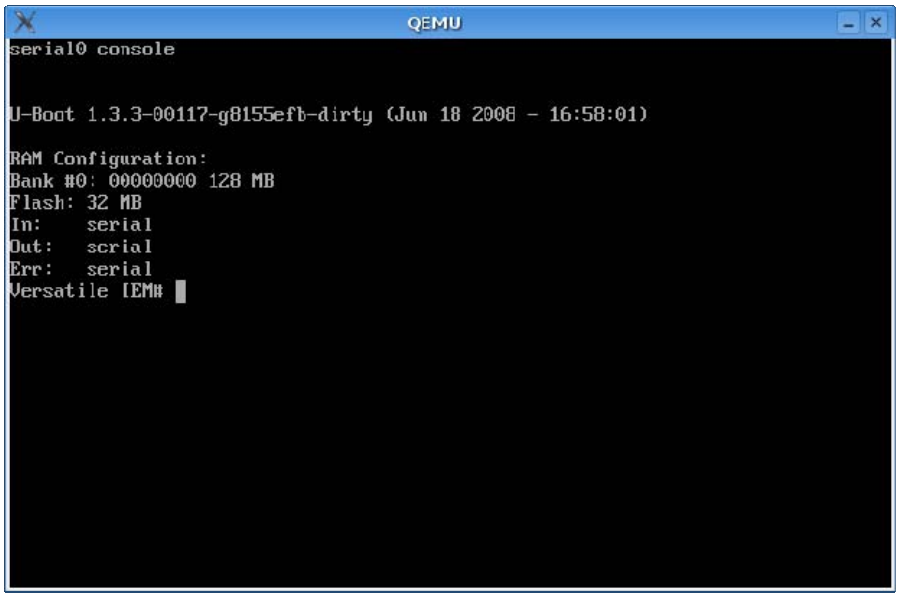


Qemu fonctionne sur quatre fenêtres différentes:

La fenêtre d'accueil (celle ci dessus) contient l'environement émulé (Ctrl-Alt 1). Pour changer de mode, il faut utiliser les touches Ctrl-Alt + touche 1 à 4:

Ctrl-Alt 2: Affiche les informations du "monitor"

Ctrl-Alt 3: Commute sur la console serie (voir ci dessous)

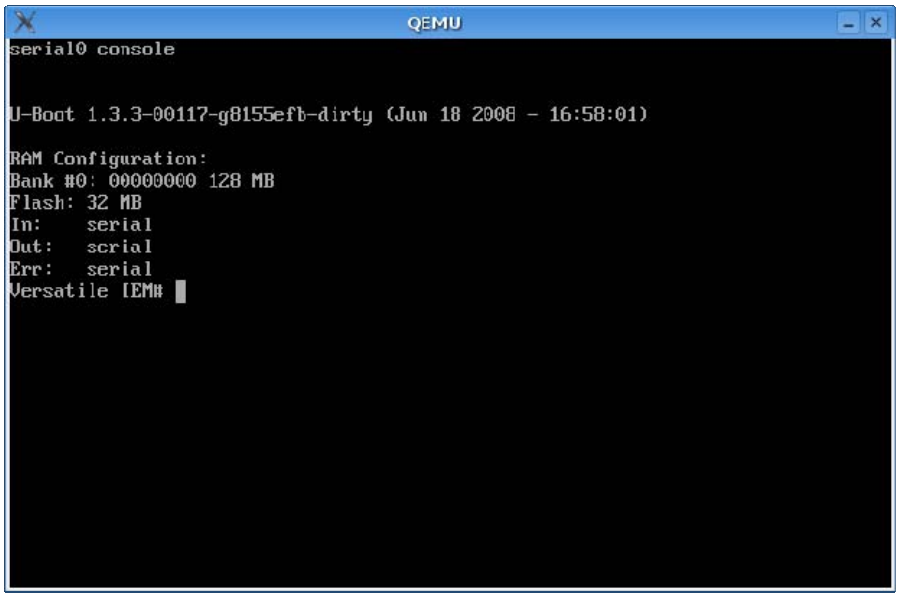


Ctrl-Alt 4: Commute sur la console parallèle

Ctrl-Alt: Sort du system émullé et redonne la mains a l'OS principal (Linux-Gentoo) ainsi que les périfiriques ci rapportant (par ex: la souris) .

7. U-boot

U-boot.bin est paramétré dans la commande qemu-system-arm. On le retrouve dans la console série de Qemu voir ci-dessous. Versatile IEM est une émmulation d'une machine ARM 926EJS.



Voici quelque commande de base:

- help (ou ?): liste et explique toute les commandes disponibles de U-boot

    # **help printenv** (explique la commande printenv)

- printenv: liste les variables d'environnement ainsi que leur valeurs.

- setenv: Ajoute ou modifie une variable d'environnement (sert a tranférer un prog)

    # **setenv *nom_variable* tftp 0x*adresse-de-destination-du-prog* /adresse/source/prog**

- run: charge un programme défini par une variable d'environnement

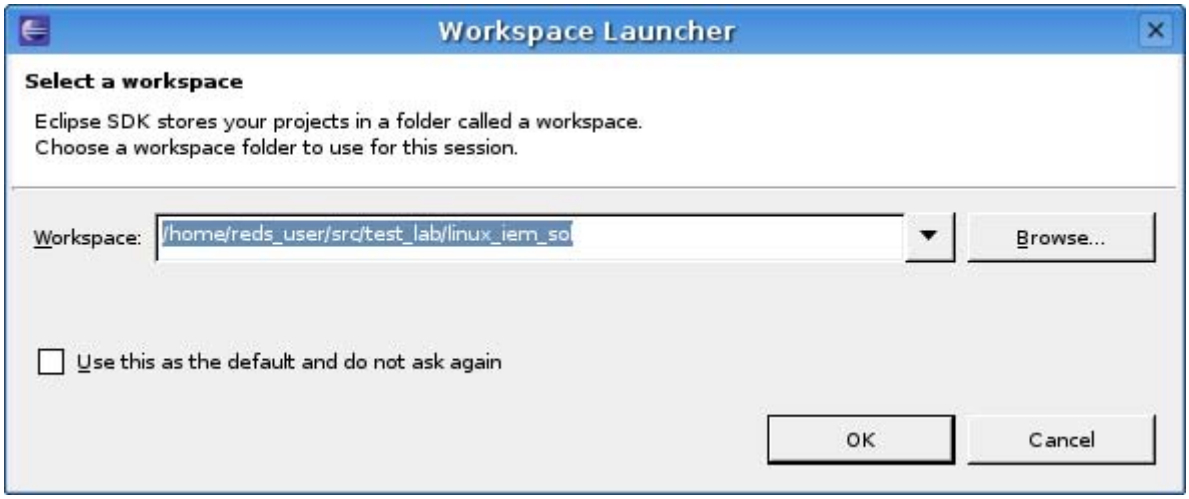
    # **run nom_variable**

- go: démmare l'application a l'adresse 0x????

    # **go 0x8000**

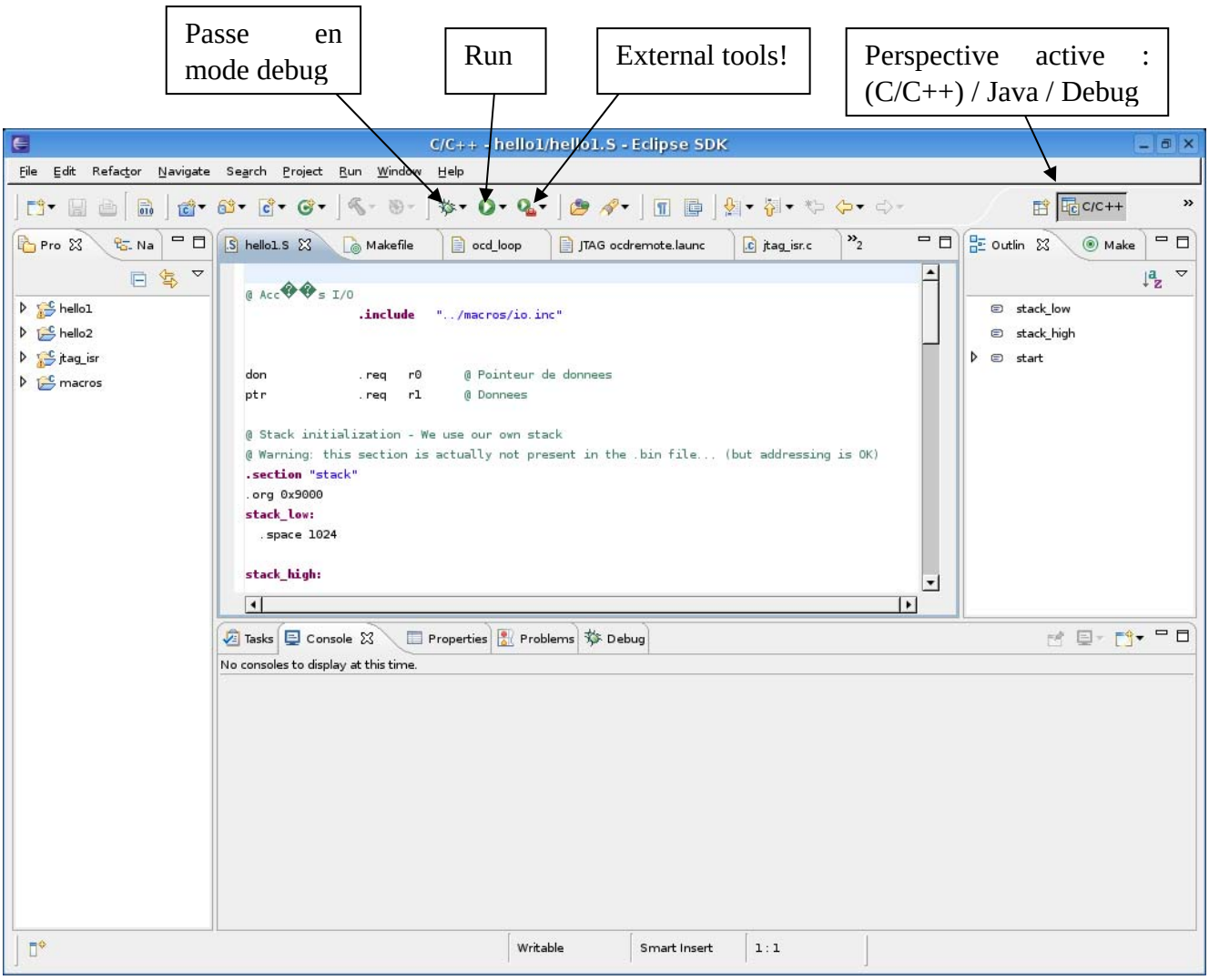
8. Debugg par JTAG USB

Pour réaliser du debug par JTAG USB sur la carte EMB du laboratoire, on utilise l'environnement de debugg *d'eclipse*. Eclipse se lance simplement en tapant "eclipse" dans une console. Remarque: vous devez-être en mode "reds_user" pour qu'eclipse démarre correctement, en "root" le system ne trouve pas la bonne JVM (java vituel machine).

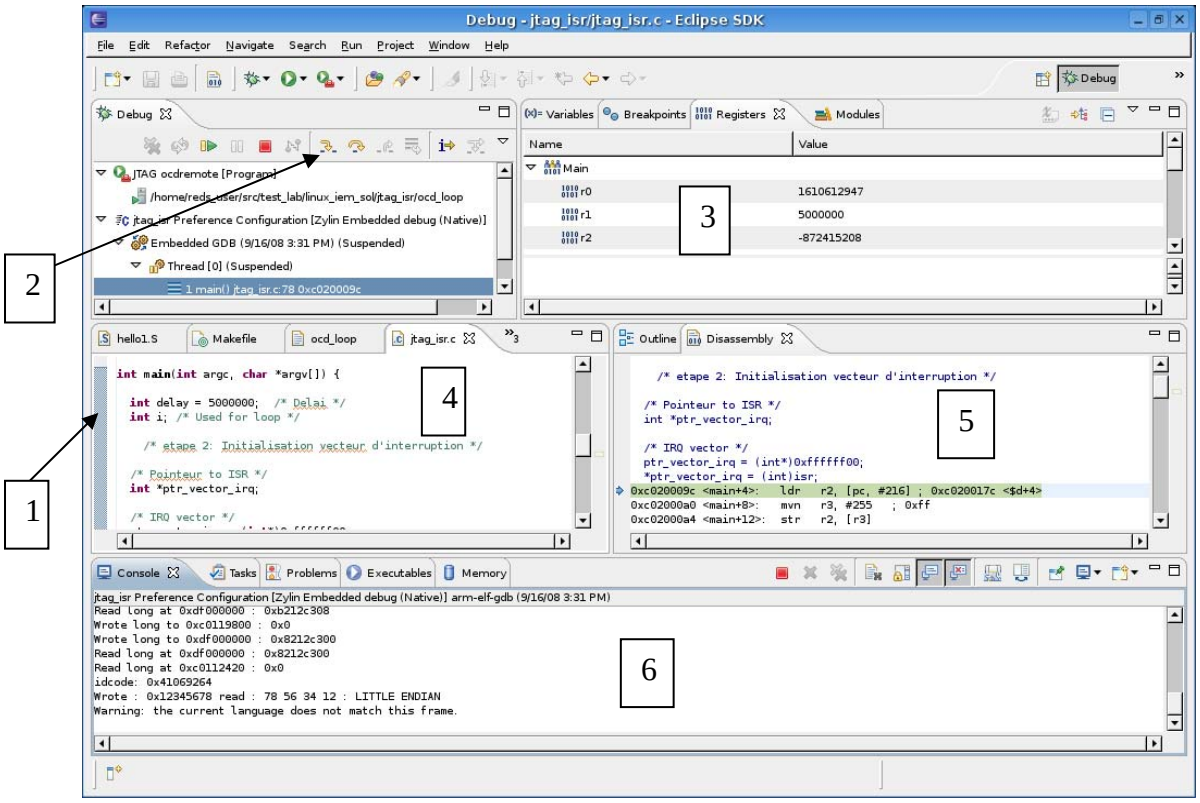

Workspace Launcher
Select a workspace
Eclipse SDK stores your projects in a folder called a workspace.
Choose a workspace folder to use for this session.
Workspace:
/home/reds_user/src/test_lab/linux_iem_so
Browse...
Use this as the default and do not ask again
OK
Cancel

Choisissez-votre espace de travail!

Info utile:



Fenêtre debug:


Debug - jtag_isr/jtag_isr.c - Eclipse SDK
File Edit Refactor Navigate Search Run Project Window Help
Debug
Variables Breakpoints Registers Modules
Name Value
Main
r0 1610612947
r1 5000000
r2 -872415208
JTAG ocdremote [Program]
/home/reds_user/src/test_lab/linux_iem_sol/jtag_isr/ocd_loop
jtag_isr Preference Configuration [Zylin Embedded debug (Native)]
Embedded GDB (9/16/08 3:31 PM) (Suspended)
Thread [0] (Suspended)
1 main() jtag_isr.c:78 0xc020009c
hello1.S Makefile ocd_loop jtag_isr.c
int main(int argc, char *argv[]) {
int delay = 5000000; /* Delai */
int i; /* Used for loop */
/* etape 2: Initialisation vecteur d'interruption */
/* Pointeur to ISR */
int *ptr_vector_irq;
/* IRQ vector */
Outline Disassembly
/* etape 2: Initialisation vecteur d'interruption */
/* Pointeur to ISR */
int *ptr_vector_irq;
/* IRQ vector */
ptr_vector_irq = (int*)0xffffff00;
*ptr_vector_irq = (int)isr;
0xc020009c <main+4>: ldr r2, [pc, #216] ; 0xc020017c <$d+4>
0xc02000a0 <main+8>: mvn r3, #255 ; 0xff
0xc02000a4 <main+12>: str r2, [r3]
Console Tasks Problems Executables Memory
jtag_isr Preference Configuration [Zylin Embedded debug (Native)] arm-elf-gdb (9/16/08 3:31 PM)
Read long at 0xdf000000 : 0xb212c308
Wrote long to 0xc0119800 : 0x0
Wrote long to 0xdf000000 : 0x8212c300
Read long at 0xdf000000 : 0x8212c300
Read long at 0xc0112420 : 0x0
idcode: 0x41069264
Wrote : 0x12345678 read : 78 56 34 12 : LITTLE ENDIAN
Warning: the current language does not match this frame.
1
2
3
4
5
6

1) Possibilité de mettre des breakpoints

2) Mode de debug (pas à pas, …)

3) Fenêtre des registres

4) et 5) Fenêtres contenant le code

6) Affichage de la console

9. Liens

Quelques lien qui peuvent vous êtes utiles :

- http://www.microcross.com/
- http://www.**eclipse**.org/